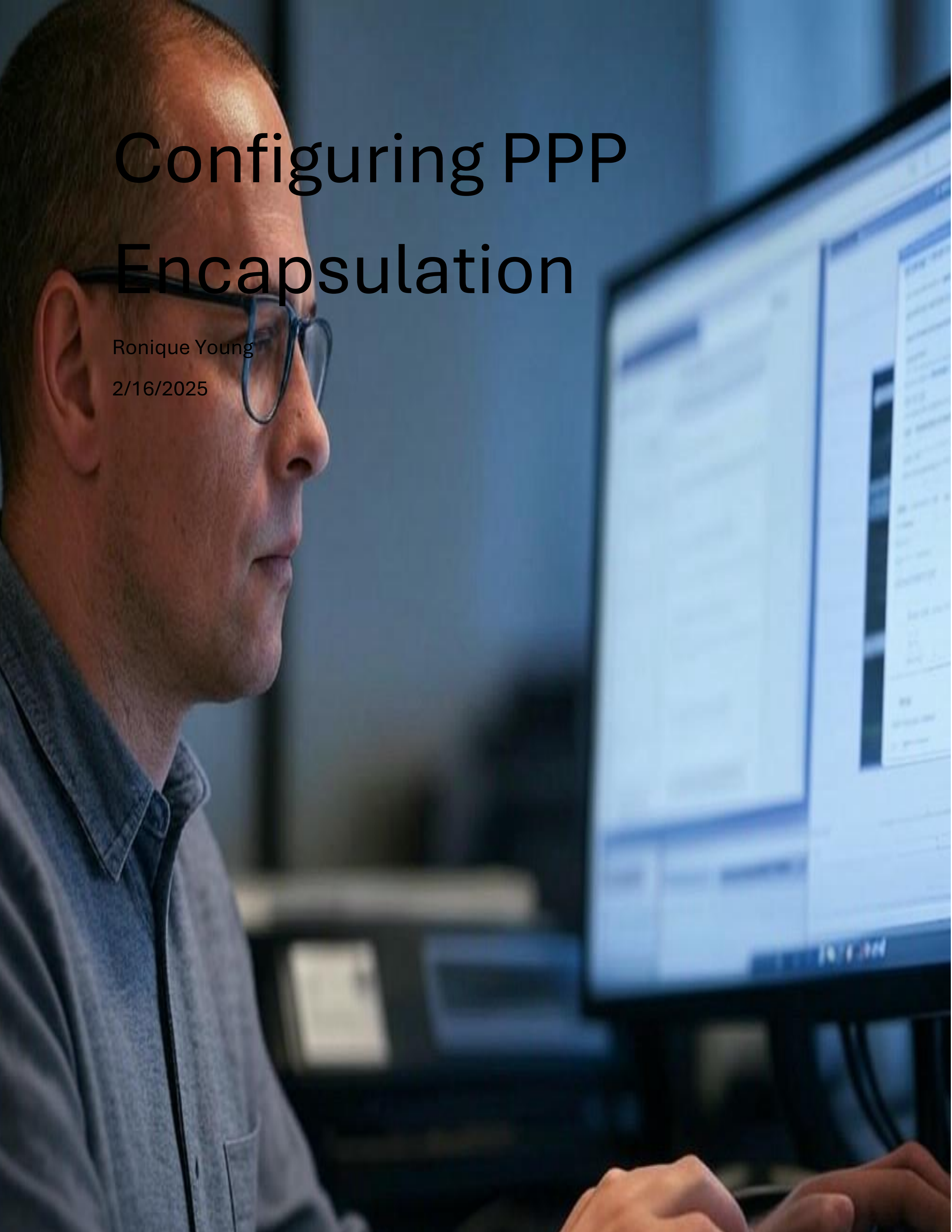


Configuring PPP Encapsulation

Ronique Young

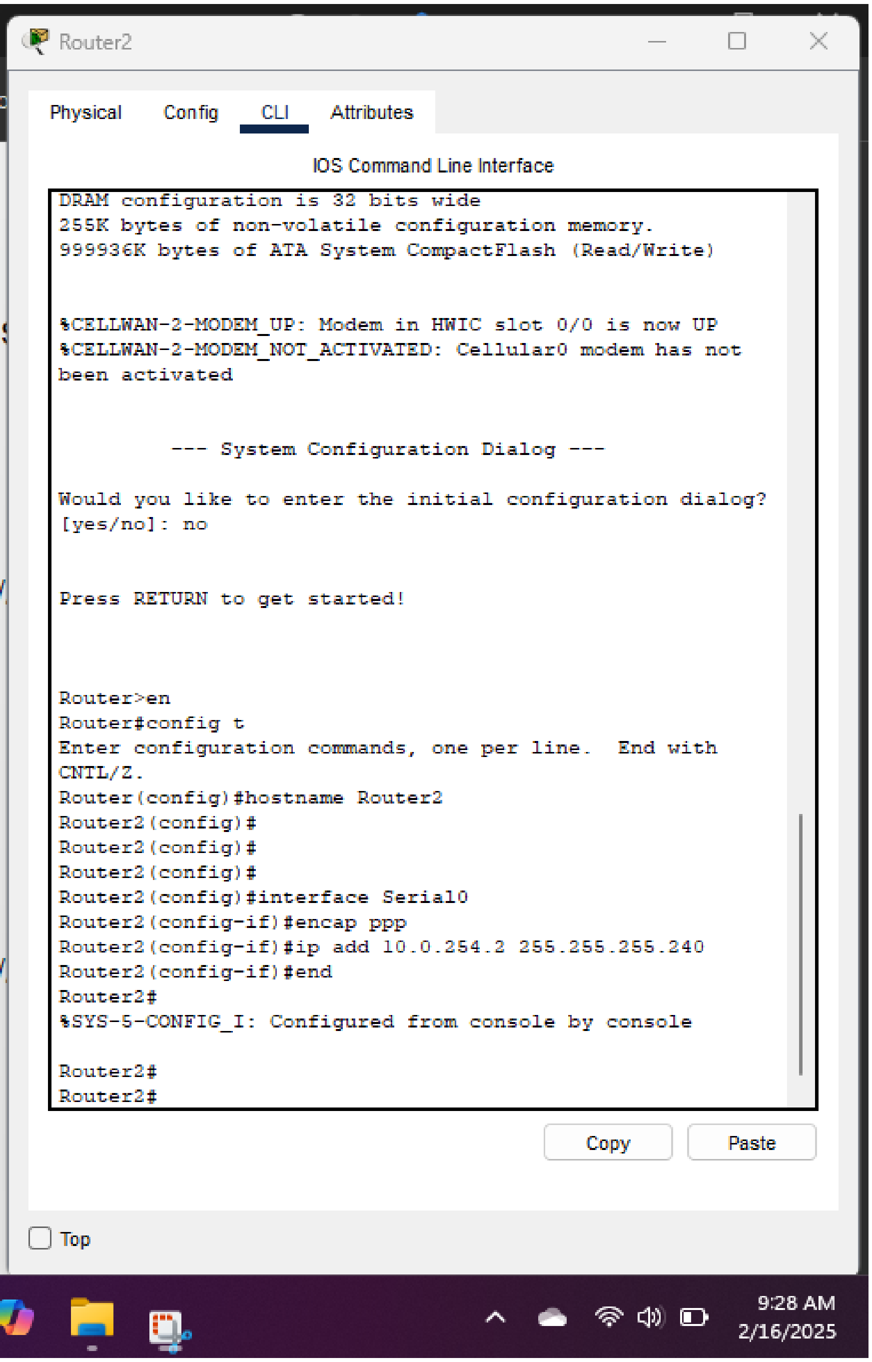
2/16/2025

In this task, I began by accessing the privileged EXEC mode on the router by typing ` Router>en ` , which stands for "enable". This step is crucial as it allows me to enter the configuration mode where I can make changes to the router's settings.

Once in the privileged mode, I entered global configuration mode using the command ` Router#config t ` , which is a shorthand for "configure terminal". This mode allows for the modification of the router's configuration. Here, I was prompted to enter configuration commands one by one, ending each session with CNTL/Z, which is a common practice for exiting configuration mode on Cisco devices.

I then changed the router's hostname from the default "Router" to "Router1" with the command ` Router(config)#hostname Router1 ` . This not only personalizes the device for easier identification in a network but also ensures that all subsequent commands reflect the new hostname.

Next, I moved to configure the Serial0 interface by typing ` Router1(config)#interface Serial0 ` . This command shifts focus to the Serial0 interface, preparing it for further configuration. I set the encapsulation protocol to PPP (Point-to-Point Protocol) with ` Router1(config-if)#encapsulation ppp ` , which is essential for establishing a direct link between two nodes in a WAN.



In this task, I started by entering privileged EXEC mode on the router with the command `Router>en`, which is shorthand for "enable". This command is necessary to access higher-level command options for configuration.

I then transitioned into global configuration mode using `Router#config t`, or "configure terminal", which allows for the manipulation of the router's settings. The system prompted me to enter configuration commands one at a time, with the option to end with CNTL/Z.

I renamed the router to "Router2" by using the command `Router(config)#hostname Router2`, which helps in identifying the device within a network environment. After setting the hostname, I moved to configure the Serial0 interface by typing `Router2(config)#interface Serial0`.

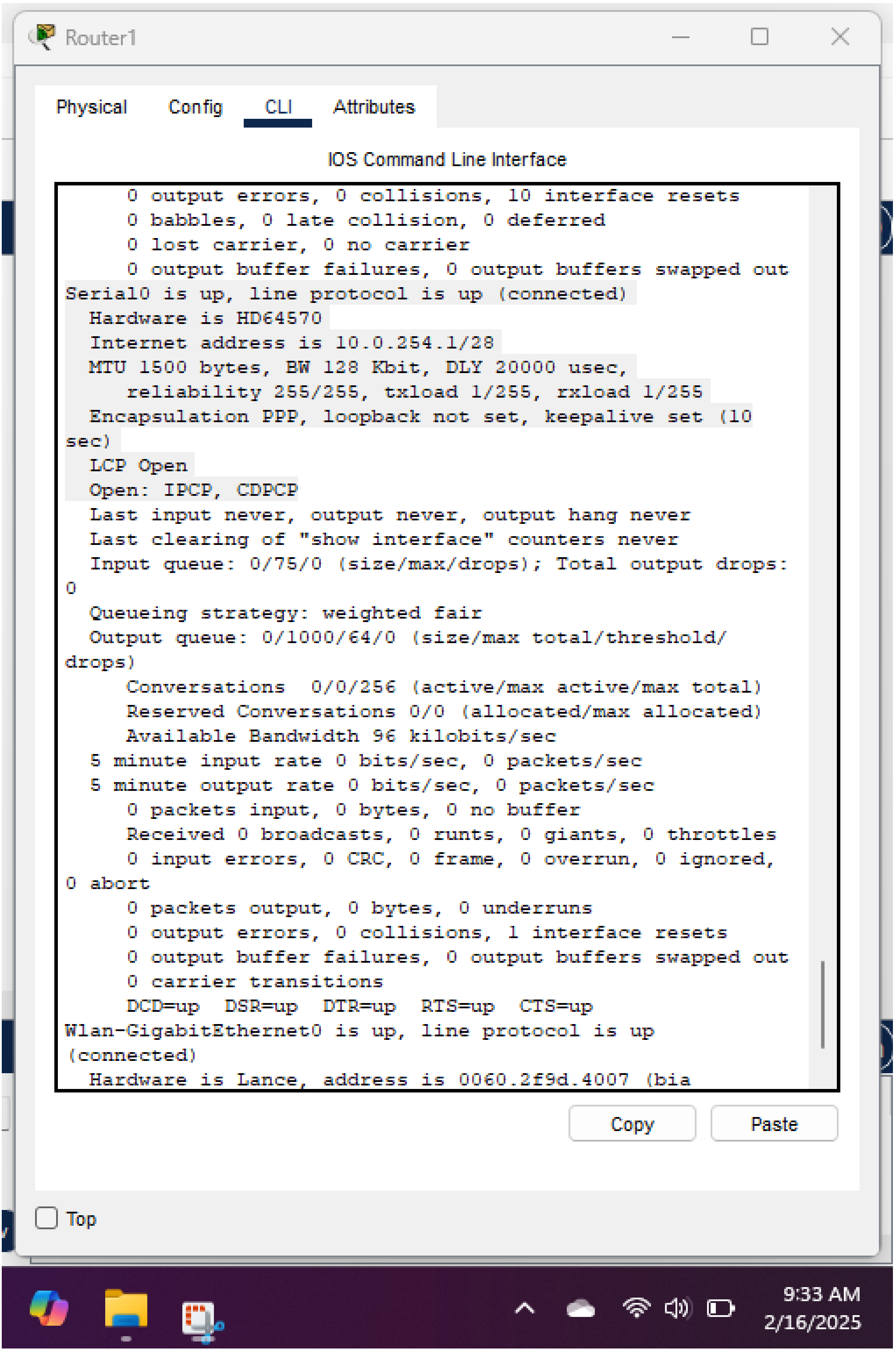
Once on the Serial0 interface, I set the encapsulation method to PPP (Point-to-Point Protocol) with the abbreviated command `Router2(config-if)#encap ppp`, which does not require the full "encapsulation" keyword, showcasing a common shortcut in Cisco IOS.

Next, I assigned an IP address to this interface using `Router2(config-if)#ip add 10.0.254.2 255.255.255.240`, which gives Serial0 the IP address 10.0.254.2 with a subnet mask that supports a small network segment.

I concluded the configuration session by typing `Router2(config-if)#end`, which exits back to the privileged EXEC mode. The router confirmed the changes with the message `%SYS-5-CONFIG_I: Configured from console by console`, indicating that the configuration was successfully applied from the console.

Following this, I assigned an IP address to the Serial0 interface with `Router1(config-if)#ip address 10.0.254.1 255.255.255.240`. This command assigns the IP address 10.0.254.1 with a subnet mask of 255.255.255.240, which provides a small network for point-to-point links.

I finished the configuration by exiting to the privileged EXEC mode using CNTL/Z (`Router1(config-if)#^Z`). The system then confirmed that the configuration was completed from the console with the message `%SYS-5-CONFIG_I: Configured from console by console`, ensuring that the changes were saved and applied.

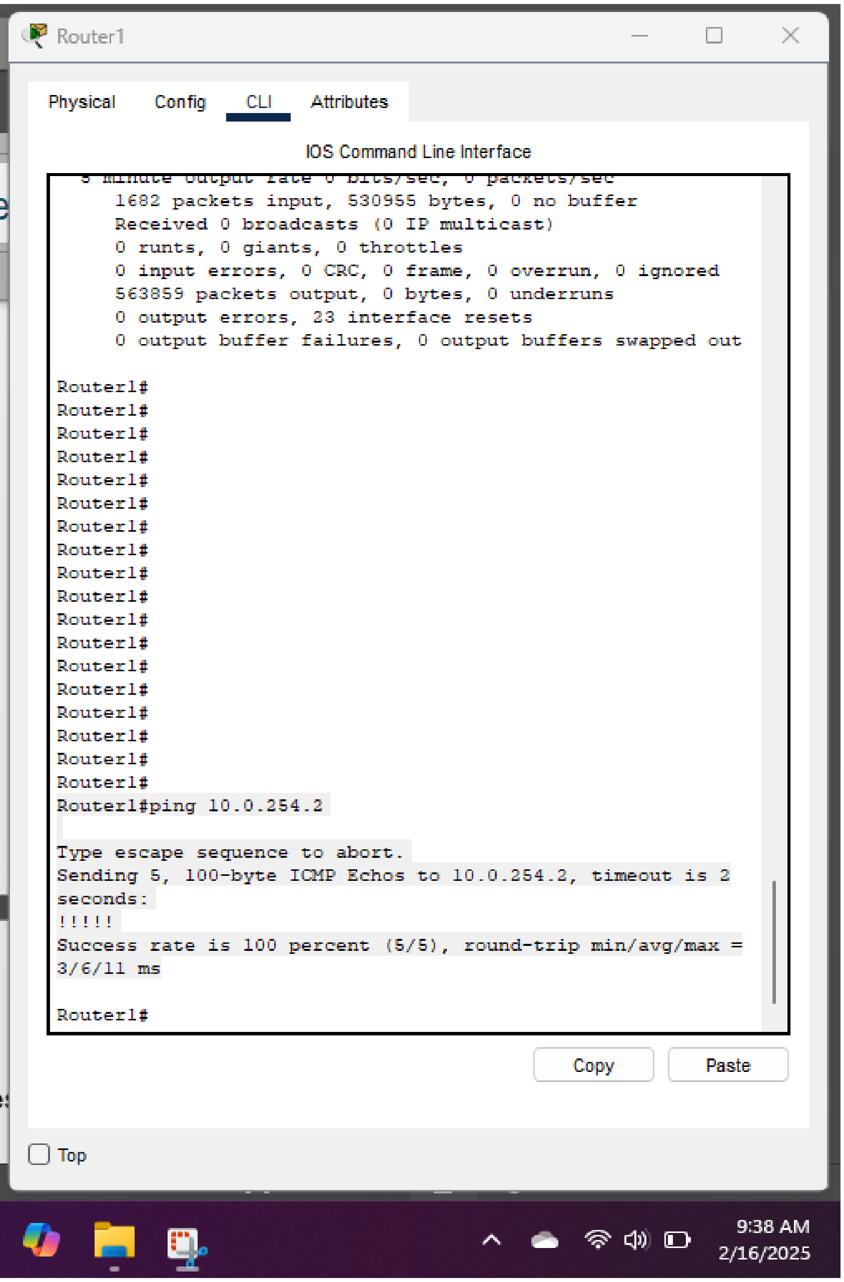


In this scenario, we're looking at the status and configuration details of the Serial0 interface on a router. The interface status indicates "Serial0 is up, line protocol is up (connected)," which means the physical layer (layer 1 in the OSI model) and the data link layer (layer 2) are both functioning correctly, and there is an active connection.

The hardware type is specified as HD64570, which is a common chip used in serial interfaces. The interface has been assigned an IP address of 10.0.254.1 with a subnet mask of /28, which aligns with the earlier configuration commands where we saw the IP address set.

The Maximum Transmission Unit (MTU) is set to 1500 bytes, which is standard for most Ethernet networks but applied here to a serial link. The bandwidth (BW) is specified as 128 Kbit, with a delay (DLY) of 20,000 microseconds, giving insights into the expected performance of this link. Both reliability and load metrics are at their best with reliability at 255/255 (perfect) and both transmit and receive loads at 1/255, indicating very low utilization.

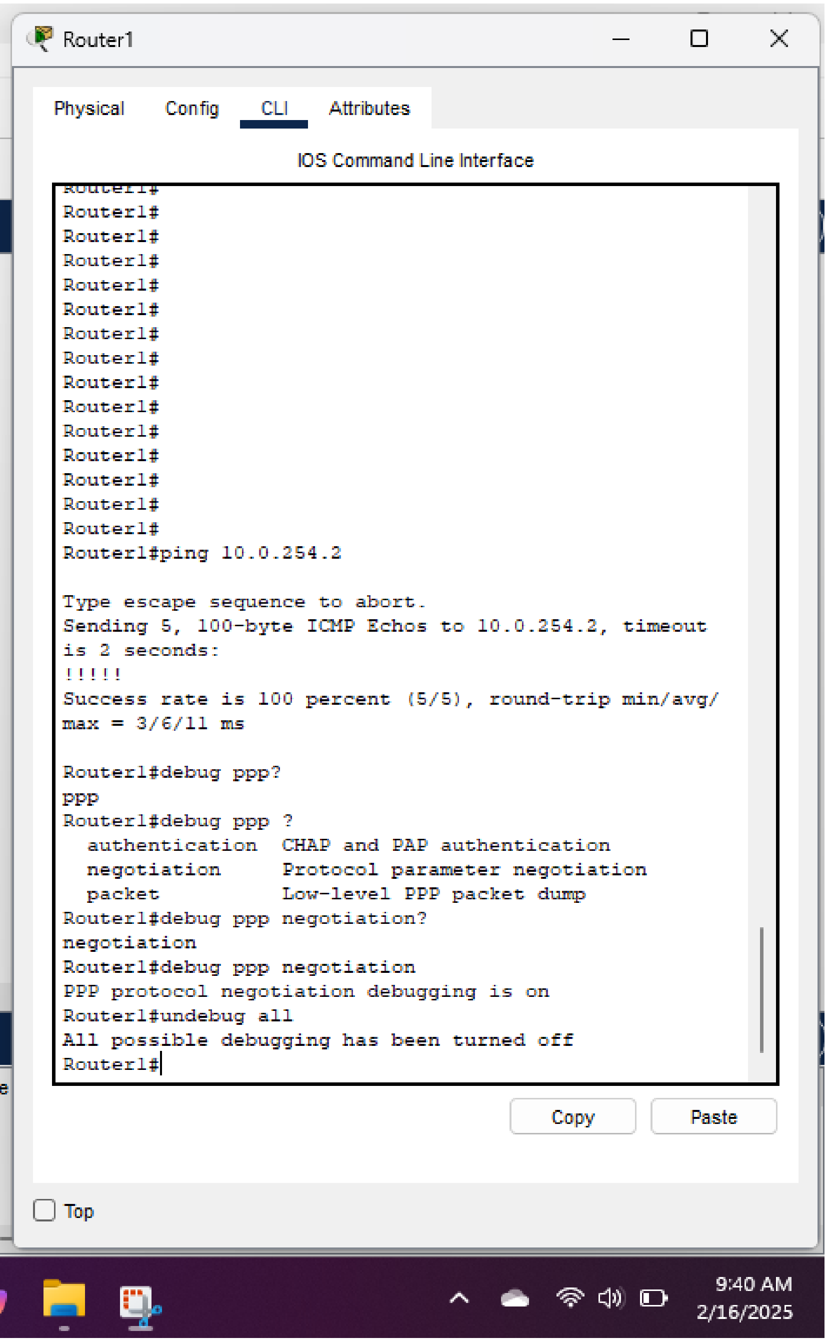
The encapsulation is set to PPP, as previously configured, and keepalives are enabled to check the link's status every 10 seconds. The status of the Link Control Protocol (LCP) is open, which means the PPP connection is established. Furthermore, both the Internet Protocol Control Protocol (IPCP) and Cisco Discovery Protocol Control Protocol (CDPCP) are also open. This suggests that IP communication and Cisco's proprietary CDP are operational over this interface, allowing for network discovery and IP packet transmission.



In this scenario, from Router1, I initiated a ping command to test connectivity to the IP address 10.0.254.2, which is presumably the address of Router2's Serial0 interface as seen in previous configurations.

The output shows that five ICMP Echo packets were sent, each 100 bytes in size, with a timeout period of 2 seconds for each echo request. The result of the ping was successful, as indicated by the "!!!!!" pattern, where each exclamation point signifies a successful echo reply.

The summary at the end states a success rate of 100 percent, meaning all five packets were replied to, which confirms that there is good connectivity between Router1 and Router2 over the Serial0 interface. The round-trip times for these packets were recorded with a minimum of 3 milliseconds, an average of 6 milliseconds, and a maximum of 11 milliseconds, indicating low latency communication between the two routers.



In this scenario, I first enabled the debugging for PPP (Point-to-Point Protocol) negotiation on Router1 by entering the command `Router1#debug ppp negotiation`. This command turns on the logging of events related to the negotiation process of PPP, which would be useful for troubleshooting PPP connections, such as seeing when LCP (Link Control Protocol) and NCP (Network Control Protocol) phases occur, or identifying any negotiation problems.

However, I then decided to disable all debugging with the command `Router1#undebug all`. This command stops all active debug processes on the router to prevent the console or logging buffer from being overwhelmed with debug information, which could be particularly useful if I only needed a temporary view of the PPP negotiation or if I was done troubleshooting. The output confirms that all possible debugging has been turned off, returning the router to a state where no debug information is being actively logged or displayed.